CSC256/MAT256: Discrete Mathematics Syllabus

Instructor

Andy Wildenberg, Tech 17A, 238 7380 (office), 839 4066 (cell)
Office hours: 10:00am-11:00pm MWF

Introduction

Discrete Mathematics is actually the first kind of mathematics that most children are exposed to in elementary schools. When we learn to count, we learn 1,2,3,4, and it takes a while to even realize that there might be anything in between those numbers. When we learn to divide, we learn that $7 \div 3 = 2R1$ and don't even seem to notice that there's something weird about the fact that division is the only operator that takes in two numbers and spits two out again. We also never seem to notice that while we're taught how to deal with negative numbers with $+, \times$, and -, we never divide by a negative number.

Even more strangely, shortly after we master how to do division, school suddenly switches to using real numbers and rational numbers suddenly that division algorithm we learn has only one answer instead of two. And by now, it's possible that you don't even remember how to do a "third grade" division algorithm anymore.

But in computer science, it turns out that the study of discrete mathematics is actually more useful than the study of real numbers that you have been taught in high school. Surprisingly it turns out that the stuff you learned out first is actually quite a bit more difficult and subtle than what you learned later. In this class we concentrate on learning all about the mathematics of the discrete; systems where you really do go from 2 to 3 with nothing in between. But this math involves a lot more than numbers. We start off with propositional and predicate logic (it's either 'true' or 'false', not half way or 0.8 'true'), move on to an introduction to number theory, number representation in different basis, and then quickly on to a study of functions where we're manipulating the functions as mathematical objects instead of just evaluating them. Finally we'll finish off with a few more advanced topics such as a brief introduction to graph theory.

The emphasis in this class will be on topics that are most relevant to computer scientists. However, this does not change the fact that this is a class on mathematics and will be treated as such. There will be a lot of

work to get you introduced to mathematical formalisms, including reading and writing precise definitions and reading and writing proofs (including proofs by induction). Having said that, if you're in this class you're already a competent programmer, and the particular skills that have allowed you to be successful with programming are in fact the same skills that you need to deal with formal defnitions and proofs.

Class hours and attendance

Students are expected to attend all classes. Please inform me as soon as you know that you will not be able to attend a session. In cases of extraordinary absenteeism, I reserve the right to alter your grade accordingly.

Grading

Grading will consist of exams, programming assignments, quizzes and presentations. There will be one midterm exam and one final exam. The midterm will count for 25

Plagiarism

It is expected that you do your own work. Obviously cheating is not acceptable. Plagiarism is a serious offense and will be treated as such. We will talk in class about what kind collaboration is acceptable, but in general, if the list of authors does not accurately state who wrote the code, plagiarism has occurred. If in doubt, please ask.

A typical penalty for cheating or plagiarism will be a zero on the assignment in question plus a two letter grade reduction in the final grade for the class.

Accommodation

If there is anyone who needs any special accommodations in class, please let me know as soon as possible

Topics

Our textbook is 'Discrete Mathematics' by Rosen. The particular edition of the book you get doesn't really matter: this is not a rapidly changing field and the homework problems will not be lifted directly from any particular edition of the book. Furthermore, the order of topics will follow the course

slides written by my former colleague Steve Skeina at SUNY Stony Brook. Those topics are:

1. Introduction to the field and course, course rules
2. Propositional Logic (Intro)
3. Logical Equivalence
4. Logical Consequence and Valid Arguments
5. Normal Forms and Logic Programming
6. Digital Logic and Circuits
7. Introduction to Functions and Set Theory
8. Properties of Set Theory: Distributivity, Associativity, etc.
9. Intro to Number Theory
10. Divisibility and Primality
11. Functions as Mathematical Objects, Inverses, 1-1, Onto, Perfect Shuffles
12. Hashing, Pigeonhole Principle
13. Recursive and Explicit Functions,
14. Functional Programming
15. List Processing in SML
16. Higher Order Functions
17. Tower of Hanoi
18. Mathematical Induction
19. Induction and Summation, Induction and Divisibility, Induction and Non-numeric Problems
20. Relations, Relational Algebra, and Relational Databases
21. Graphs
22. Trees

Objectives

Upon the successful completion of this class, students will be able to

1. write truth tables and propositional logic proofs involving arbitrary numbers of free variables and assumptions
2. read and write mathematical definitions and proofs for basic mathematical concepts in number theory, set theory and graph theory.
3. write functions that handle list processing in SML
4. be able to manipulate functions as first class mathematical entities
5. provide correct mathematical definitions for following kinds of numbers: prime/composite, odd/even , natural/integers/rationals